I.J. Information Technology and Computer Science, **2011, 1, 1-9**
Published Online February 2011 in MECS (http://www.mecs-press.org/)

A Hybrid P2P Approach to Service Discovery in the Cloud

Jing Zhou
Communication University of China, Beijing, China
Chinese Academy of Sciences, Beijing, China
Email: zhoujing@cuc.edu.cn

Nor Aniza Abdullah
University of Malaya, Kuala Lumpur, Malaysia
Email: noraniza@um.edu.my

Zhongzhi Shi
Chinese Academy of Sciences, Beijing, China
Email: {shizz}@ics.ict.ac.cn

***Abstract*—Highly scalable techniques for service discovery are key to the efficient use of Cloud resources, since the Cloud computing appears to be part of the mainstream computing in a few years. We embarked on a preliminary study on Cloud service discovery by adopting an unstructured P2P paradigm. We developed an efficient mechanism for routing of service requests by coupling a number of components: one-hop replication, semantic-aware message routing, topology reorganization, and supernodes. A number of experiments were carried out that demonstrated the expected performance of the proposed P2P search scheme.**

***Index Terms*—performance, service discovery, the Cloud, unstructured P2P paradigm**

I. Introduction

Cloud computing [1] comes in three kinds in terms of the services it supplies: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [2]. Currently, users can access Cloud services (offered by major providers such as Amazon, Microsoft, Google App Engine, Eucalyptus, and GoGrid) by visiting the service provider's website, establishing a runtime environment in response to instructions, creating a user account, configuring related tools, and writing a few lines of code if required. No complicated mechanism for service discovery is necessarily involved. As the Cloud computing appears to become part of the mainstream computing in a few years, the number of the services it provides, its users, and the requests of these services will increase in orders of magnitude. We are convinced that highly scalable techniques for service discovery are key to the efficient use of Cloud resources.

Observations on various distributed systems including Grid computing show that service discovery mechanisms are primarily centralized. It is well known that centralized service discovery gives rise to the following issues: 1) single point of failure; 2) lack of satisfactory scalability; 3) requirement of powerful computing capabilities to serve large amounts of service discovery and update queries on the central server; and 4) performance bottlenecks and network congestion.

To overcome the drawbacks and inefficiencies of centralized service discovery in terms of scalability, fault-tolerance, and network congestion, a number of fully decentralized solutions were proposed. Among others, the approach that adopts the P2P paradigm draws a great deal of attention. The characteristic features of P2P systems comprise self-organization, fault tolerance, and scalability that make P2P computing an obvious candidate for addressing large scale service discovery in the Cloud environment.

Studies [3] showed that satisfactory scalability can be achieved if P2P techniques based on DHTs (distributed hash tables) are applied to service discovery in Cloud computing. We, however, argue that such structured P2P techniques place severe constrains on the network topology and the placement of services (or their indices), which makes structured P2P less capable of modeling the real world (see Sect. II-B). We are therefore in favor of unstructured P2P techniques to address the problem of interest.

We assume that all the Cloud service providers wishing to share their services and the Cloud service requesters voluntarily form a P2P network. Upon arrival, each peer will exchange the information about the Cloud services it hosts with neighboring peers, that is, one-hop replication [4] is carried out. We set out to address the scalable mechanism for service discovery in such a network by utilizing Semantic Web technologies to describe Cloud services in a semantic manner (Sect. III-A). Considering the multi-dimensional property of both the semantic descriptions of Cloud services and the user requests for the services, we employed the kd-tree to establish a data space, that is, to create indices, for the description of all the services that any peer is aware of (Sect. III-B) and developed a hybrid P2P search scheme to efficiently find services of interest that combines one-

hop replication, semantic-aware routing, shallow flooding, topology re-organization, and supernodes (Sect. III-D).

Our experience with unstructured P2P without any form of central authority [5] [6] [7] reveals that a certain degree of centralization (supernodes for instance) can help strike a good balance between information freshness and system scalability. We therefore allowed a small number of "knowledgeable" nodes that maintain the service information of a group of other nodes to elect themselves to be supernodes at any given time[1]. These supernodes broadcast their status to neighbors and routinely probe the latter for their up-to-date information.

The major contribution of our work consists of the following.

- Providing good support for both point and range queries: Thanks to the use of the kd-tree, both a point query and a range query can be similarly handled in our scheme. This is in contrast to other approaches in which resolving a point query is easy but supporting range queries requires sufficient extensions to the basic mechanism (DHTs in structured P2P for instance).
- Developing a localized search scheme: We proposed a localized scheme when constructing the kd-tree index for each peer node, that is, no global knowledge is required. However, in [9] when "skip pointers" (shortcuts used for optimized routing based on kd-trees) are built, the set of all nodes should be known in advance.
- Employing semantic-aware routing protocols and topology reconstruction techniques: When forwarding a service request, a peer node decides the next-hop node for the request message in response to the semantic relationship between the service request and the service information of its neighbors. Moreover, a node proactively seeks new neighbors that would satisfy the service requests it sends in the future with high probability. As a result, the P2P network topology is reformulated.

The remainder of the paper is structured as follows. Related work is reviewed in Sect. II. We present and discuss primary design issues in Sect. III. This is followed by a detailed description of our experiments in Sect. IV. Finally, we conclude the paper by identifying open issues in Sect. V.

II. RELATED WORK

A. Service Description and Discovery Using Semantics

A service description is a specification of the functional and non-functional capabilities and characteristics of a service. When describing a service, the service properties, capabilities, and constraints should all be taken into account. In the real world, the service providers tend to describe devices in terms of lower level properties whilst the service requesters usually prefer to describe service requests using more abstract or higher level concepts.

[1]This is similar to the practice in wireless sensor networks [8].

The Semantic Web offers a number of powerful tools and can therefore service such a request well. The integration of the Semantic Web technologies is beneficial to the realization of efficient service discovery due to the following reasons:

- Ontology can be used to describe concepts and the relationships among the concepts within a specific domain in a disambiguous way;
- Languages such as OWL (the Web Ontology Language) can be used to describe ontologies in order to support semantic inference for relationships among various concepts;
- A number of tools in the Semantic Web community have been developed for service (Web services in particular) description purposes, including OWL-S [10], WSMO (Web Services Modeling Ontology) [11], and Web Service Semantics–WSDL-S [12];
- The semantic service description with powerful expressiveness is necessary to service matching based on semantics.

Semantic service discovery is typically carried out by seeking the service of interest according to service capabilities. Apart from the language that describes service capabilities, an algorithm specification for matching between service requests and service descriptions is also required. Semantics-based service matching comprises signature matching [13] and specification matching [14], and has been applied to a number of distributed systems, including Grid computing [15], [16], P2P computing [17], [18], Web services [19], and pervasive computing [20].

Zeng *et al.* proposed a matching algorithm based on WordNet in [21] to address the issue of service matching in the Cloud environment. The basic idea is to extend the keywords that describe the input capabilities and output capabilities of services by using WordNet. A function is then employed to evaluate the semantic similarity between the concept sets (consisting of the concepts extended from the aforementioned keywords) of any two services. The primary disadvantage of the approach is lack of support for range queries.

B. Unstructured P2P and structured P2P

Unstructured P2P systems, including Gnutella, Freenet [22], FastTrack, and Kazaa, carry out object lookup and downloading operations in the absence of a central index server. Each peer maintains indices for the resources it currently holds. A lookup operation in such systems will not necessarily be successful and there is no upper and lower bounds on the successful operation. Furthermore, unstructured P2P supports one-dimensional and multi-dimensional point queries and range queries.

Similarly, there is no central index server in structured P2P systems based on DHTs (such as CAN [23], Chord [24], Pastry [25], and Tapestry [26]). Each peer in structured P2P systems based on DHTs maintains the indices of data items on another $O(\log n)$ peers (but not the data items of its own!) where n is the number of peers in the system. The keys of the data items and nodes are mapped onto the overlay network in which each node is responsible for managing a small number of data items.

Whenever a node in the overlay network receives a lookup request, it locates the data item of interest within $O(\log n)$ hops. Therefore, the query performance delivered by structured P2P systems is deterministic. One-dimensional point queries can be well supported. However, for these systems to efficiently support one-dimensional range queries, and multi-dimensional point queries and range queries, various spatial indices [27] should be employed (such as the Space Filling Curve, k-d trees, and MX-CIF quadtree) and sufficient extensions to the basic DHT mechanism should be carried out [28].

We opt for an unstructured P2P paradigm for our service discovery scheme based upon the following considerations. In the Cloud environment, each service provider maintains and manages its own services, describing the services, creating indices according to service descriptions, and publishing service description to facilitate service discovery. As in Grid computing, we argue that service discovery in Cloud computing will also be performed on the basis of multiple (instead of single) attributes of the service in most cases. In contrast to structured P2P, the unstructured P2P paradigm can better model the real world problem. Moreover, the search mechanism enabled by unstructured P2P techniques can better resolve multi-dimensional queries by incorporating support for semantics, whereas the basic DHT technique delivers satisfactory performance only in keyword search.

C. P2P-based Service Discovery

A decentralized index system, in which the appropriate data structure is selected for building indices upon the service description, is essential for implementing a P2P-based mechanism for service discovery. Since a service is typically characterized by both static and dynamic attributes, users can specify the requirement for multiple attributes in a query, that is, a multi-dimensional query is allowed. In unstructured P2P, each peer maintains its own services and service descriptions, and even simple indices, such as the two dimensional table, can be employed to facilitate resolving multi-dimensional queries.

Additionally, the routing mechanism for multi-dimensional queries in the P2P paradigm should be developed to help forward service requests to their potential destination peer efficiently. Flooding and its variants [29] [30] are the widely used approaches to messaging routing in unstructured P2P. The main drawback of such techniques is that excessive unnecessary query messages and traffic are generated. Therefore, various heuristics [31] [7] were developed to guide message routing as well as to increase system scalability. We provide greater details in the following section.

D. Hybrid P2P Search Schemes

Typically, efficient search mechanisms combine multiple techniques together. Among others, random walk, owing to its good scalability and excellent granularity, is sometimes suggested as an alternative search technique to flooding, or are collectively applied with flooding at other times.

Chawathe *et al.* presented Gia in [31] to improve the scalability of Gnutella-like P2P systems. The major components of Gia include dynamic topology adaptation, active flow control, one-hop replication of pointers to content, and biased random walk-based search. In particular, the topology adaption algorithm ensures that high capacity nodes are ones with high degree and low capacity nodes are within short reach of higher capacity ones. Topology adaption, coupled with keeping pointers to content of immediate neighbors, enables high capacity nodes to provide answers to a greater number of queries. Gia was demonstrated to provide 3 to 5 orders of magnitude improvement in the total capacity of the resultant system which exhibits significant robustness to failures.

The rationale behind [7] is to forward resource requests in an unstructured P2P triplestore based on semantics and mediate the message routing by semantic-aware topology reorganization for further performance enhancement. The search mechanism combines local, shallow flooding with semantic-directed multiple random walk. This is one of the few work on unstructured P2P search mechanisms that take into account the semantics of both the resource request and the resource description for message routing and topology reconstruction. By contrast, node capacity is an important metric in Gia (see above) and others for topology adaptation. Simulation results indicated that the introduction of semantics leads to a search mechanism that outperforms multiple random walk, constrained flooding, and interest based locality to deliver desired scalability whilst incurring the least system load.

Gkantsidis *et al.* analytically justify that a short random walk along with shallow flooding on every step delivers particularly good performance. Moreover, normalized flooding, which allows a vertex of small degree to forward a query to all its neighbors and a vertex of large degree to propagate a query to a small subset of its neighbors uniformly at random, was proposed to rectify the deteriorated performance of flooding in the case of a sparse network with a few vertices of large degrees [32].

III. SYSTEM DESIGN

A. Service Description

We started by using WSDL-S, currently a W3C member submission, that offers a lightweight solution to the creation of semantic service description. In WSDL-S, the expressivity of WSDL was augmented with semantics by adopting concepts similar to those in OWL-S. We may gradually extend and enhance WSDL-S by means of the extensibility provided by WSDL itself according to the specific requirements for Cloud service descriptions.

In summary, WSDL-S provides a few extensibility elements to realize the URI reference mechanism as follows [12].

1) **wssem:modelReference** specifies the association between a WSDL entity and a concept in a semantic model.
2) **wssem:schemaMapping** handles the structural difference between the schema elements of a Web service and their corresponding concepts in a

```
<?xml version="1.0" encoding="iso-8859-1"?>
<definitions name="DiskStorage" targetNamespace="http://www.example.com/wsdl-s/examples/diskStorage.wsdl"
    xmlns="http://www.w3.org/2004/08/wsdl" xmlns:tns="http://www.example.com/wsdl-s/examples/diskStorage.wsdl"
    xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsd1="http://www.example.com/wsdl-s/examples/diskStorage.wsdl"
    xmlns:wssem="http://www.example.com/wsdl-s/examples/diskStorage.wsdl"
    xmlns:DSOntology="http://www.example.com/wsdl-s/ontologies/DiskStorage.owl">
    <types>
                ...
    </types>
    <interface name="DiskStorage">
        <operation name="processDiskStorage" pattern="wsdl:in-out"
            wssem:modelReference="DSOntology#RequestDiskStorage">
            <input messageLabel ="processDiskStorageRequest" element="tns:processDiskStorageRequest"/>
            <output messageLabel ="processDiskStorageResponse" element="processDiskStorageResponse"/>
        <!–Precondition and effect are added as extensible elements on an operation–>
            <wssem:precondition name="AvailableStoragePrecond"
                wssem:modelReference="DSOntology#StroageAvailable"
                expression="DSOntology#Capacity=200GB&&DSOntology#Transferrate=50Mb/s"/>
            <wssem:effect name="StorageOccupiedEffect"
                wssem:modelReference="DSOntology#StorageOccupied"/>
        </operation>
    </interface>
</definitions>
```

Figure 1 A segment of an example Cloud service description

semantic model.

3) **wssem:precondition** and **wssem:effect** are specified as child elements of the element operation and describe the semantics of the operation.
4) **wssem:serviceCategorization** comprises service categorization information that could be used when publishing a service in a Web Services registry.

A Cloud service can be semantically annotated by borrowing all these constructs from WSDL-S. Fig. 1 presents a segment of an example Cloud service description. Note that a precondition is a set of statements that should be true before a service can be successfully invoked. Hence, we can describe part of a user's requirement for a service by means of preconditions.

For instance, if a user is looking for a disk storage service, she can send out a "DiskStorage" request and specify the capacity should be 200GB and the transfer rate of the disk should be 50Mb/s. However, in proposal [12] at most one precondition (as well as one effect) is allowed so multiple preconditions should be captured into one high level precondition. We can combine those two statements via "AND" (that is, "&&" in Fig. 1) into one precondition since **WSSemantics.xsd** defines an attribute "expression" of type "string" for **wssem:precondition**.

B. Description Indexing

Among all the constructs used to describe a Cloud service, the precondition is the most important resource upon which we can build indices for all the services in a P2P network. Suppose a user issues a Cloud service request and we can then compile part of the request (which is specified in **wssem:precondition**) as follows.

((attri_1, value_1), (attri_2, value_2), …, (attri_i, value_i))

As such, a user request can be converted to a multidimensional query by which the user specifies multiple conditions should be met on several attributes. We adapted one of the database approaches—the kd-tree—to establish a data space for all service descriptions. Once the data space is constructed[2], we set out to develop the routing strategy (in the following section) by which a multi-dimensional query can be efficiently forwarded to the relevant peers in such a space.

The kd-tree is a data structure that decomposes a multidimensional data space into hyperrectangles and each node corresponds to a hyperrectangle. The fields of a kd-tree node include the splitting dimension number, splitting value, left kd-tree, and right kd-tree. According to the first two fields, each node splits the space into two subspaces. Searching for a point in the dataset represented in a kd-tree can be carried out in a traversal of the tree from root to leaf ($O(\log n)$ if there are n data points). In the following, we demonstrate a kd-tree representation of 4 points (200GB, 50Mb/s), (300GB, 80Mb/s), (600GB, 30Mb/s), and (800GB, 90Mb/s) in the multi-dimensional space. For simplicity, a 2-d space is used for illustration.

In Fig. 2, the root node with (200GB, 50Mb/s) splits the space in the y-axis into two subspaces. The point (600GB, 30Mb/s) lies in the lower space, that is, {(x, y) | y <50} and hence is in the subtree of the root node. Through the same mechanism, the descriptions of all the services hosted by peers can be mapped onto points in the multi-dimensional space which then together form a kd-tree. By traversing such a kd-tree, the hyperrectangle leaf that potentially contains the target point can be located.

[2] Note that all peer nodes incorporate their query results into the local kd-tree indices over time.

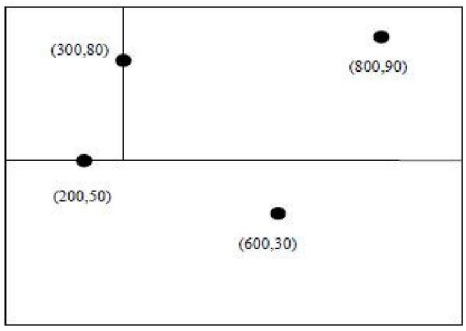

Figure 2 A kd-tree example

C. Construction and Operation of the P2P network

We assume that all the Cloud service providers willing to share their services and the Cloud service requesters (inclusive of potential free-riders) voluntarily form a P2P network. Upon arrival, each peer exchanges its information about the Cloud services it hosts with neighboring peers, that is, one-hop replication is carried out. It was reported in [32] that such replication in sparse networks yields low network overhead and its benefits can be enjoyed by all future searches.

If a peer node becomes "knowledgeable enough" in the services that other neighbors host, that is, the number of its neighbors exceeds a threshold, and considers itself capable of dealing with a large number of incoming queries, the node can elect itself supernode and notify all neighbors of its updated status. According to the routing algorithm depicted in Fig. 3, queries to be forwarded will always bias towards such supernodes for a potential shortcut to target nodes. A supernode may also reverse to an ordinary peer node at any time if it is no longer willing to take on its role by notifying all neighbors of its updated status.

```
Upon the receipt of an incoming query, peer node p0 checks
    its local kd-tree index for any match;
Results are sent back from p0 to the query originator;
The TTL of the query message is decreased by 1.
if the TTL of the message is greater than 0, then
    if p0 has service descriptions with other neighbors p_comm
        in common and the overlap contains the search terms
        in the query, then
        p0 forwards the query to each node in p_comm;
        if p0 has neighbors p_super that claim to be supernodes, then
            p0 forwards the query to each node in p_super;
    else p0 propagates the query to every neighbor;
    end_if
end_if
```

Figure 3 Query resolving and routing at query router (including query originator) p_0

Proactively, a peer will notify all neighbors of its departure from the network. In case of abrupt failure or dysfunction of a peer, the routine probing launched by its neighbors will help identify the peer disconnection. In all the above cases, information about services hosted by the leaving peer will simply be pruned from neighbors' indices.

Furthermore, peers are allowed to replace old "inefficient" neighbors with new "efficient" ones, thus leading to the change of the network topology. We defer the discussions on this topic to Sect. III-E.

D. Routing of Service Requests

Part of a user's service request, which is encoded in **wssem:precondition**, is used to formulate a multidimensional query. A multi-dimensional query Я currently supported takes the following form:

$$(\alpha\_1\ \theta\_1\ v\_1) \cap (\alpha\_2\ \theta\_2\ v\_2) \cap \ldots \cap (\alpha\_i\ \theta\_i\ v\_i)$$

where α_i refers to some attribute of the service(s) being sought, v_i specifies a value related to α_i, and θ_i denotes a relational operator that tests some kind of relation between two entities, that is, α_i and v_i in this case.

To execute such a multi-dimensional query over a P2P network, the query must be routed to the set of nodes that contain data relevant to the query. We describe the routing algorithm in Fig. 3.

Each query message is attached a TTL (Time-To-Live) tag at their originators so as to limit the propagation scope of the message. According to the content of the query, the query originator (or a peer that receives the query), say p_0, will first traverse its kd-tree to locate the set of potential target nodes. Any result will be directly sent back to the query originator if necessary. The TTL of the message is decreased by one. If the current value of TTL is equal to zero, the message is simply dropped.

If p_0 happens to have service descriptions with other neighbors in common and the overlap contains all the terms in the query, then it sends the query to each of such neighbors. This is because, as we will shortly present in the following section, peers are allowed to drop inefficient links to old neighbors whilst establishing efficient links to new neighbors. Over time, clusters that consist of peers sharing similar service descriptions will eventually emerge. Once we discover a node that satisfies the query, other target nodes can be easily located by only allowing message forwarding along efficient links that lead to peers within the same cluster.

Otherwise, if p_0 has any neighbors that are supernodes, it will always forward the query to all of such neighbor. The reason is rather simple: supernodes may find the target nodes more rapidly since they are typically more knowledgeable than others and therefore are able to provide "shortcuts" to those nodes.

Under certain circumstances, p_0 may have no overlap in service descriptions with any neighbor, so it simply floods the query to every neighbor in hope of a match. As shown by experimental results in Table I, we can cut down half of the system load (measured by *msgs*/*query* and *msgs*/*node*) at the cost of a little decreased recall level by halving the flooding message coverage in the hybrid search scheme.

Moreover, a query will only be forwarded to the same neighboring peer once. If a peer node has received multiple copies of a query message from different

neighbors, the node only sends the reply back to the query initiator or routes the query message to other neighbors in response to the first copy of the query message.

E. Re-organization of Network Topology

Unstructured P2P, the implementation of the Gnutella protocol for instance, allows peer nodes to abandon inefficient links to old neighbors and to build up efficient links to new neighbors by constantly evaluating the "efficiency" of each link3 a peer possesses, thus leading to the re-organization of the network topology with the primary objective to improve the performance of searching [32]. We adopt the same concept but make use of different evaluation metrics to quantify the efficiency of links to neighbors.

It is obvious that "semantics" has been paid extra attention to both in the design of service description and service request routing. Similarly, when peers seek more efficient links (to new neighbors), they will take into account the semantic relationship between service descriptions provided by themselves and service descriptions provided by candidate peer nodes. The more overlap, the more efficient the links to neighbors. Since the number of neighbors that each peer has should not exceed a threshold, replacement of old neighbors by new ones occurs regularly.

Topology re-organization will gradually lead to "clusters" within which peers share service descriptions in common. Recall the routing algorithm for service requests in Sect. III-D. Within a cluster, if one peer that possesses semantically related service descriptions with the service requests, other targets can be easily located by forwarding the query message along the links that relate the peer and those targets.

We only consider in this work the topology re-organization voluntarily launched by peers. Topology re-construction caused by peer departure as described in Sect. III-C is out of the scope of the paper.

IV. EXPERIMENTS

We present in this section our experimental evaluation of the performance of the proposed hybrid P2P search scheme. Flooding and (multiple) random walk searching techniques were used as benchmark for comparison with the scheme. The experimental results quantify the performance of the proposed hybrid heuristics.

A. Settings and Evaluation Metrics

We carried out a series of simulations on the hybrid P2P search scheme and compared its performance with constrained flooding and multiple random walk by varying each of the following parameters to simulate various network conditions:

- *Search method*: the way that service queries are forwarded and served within a network.
- *One hop replication*: boolean value that specifies whether peer nodes replicate their neighbors' service information.
- *Result caching*: boolean value that indicates whether peer nodes check local cached query results when resolving incoming queries.
- *Number of supernodes*: the number of supernodes in the in silico P2P network.
- *Number of walkers*: the number of walkers in multiple random walk.

The topology of the P2P network (see Sect. III-C) was generated by use of the network topology generator Inet 3.0 [33]. During the experiments on topology reorganization, this initial topology was altered wherever necessary. For constrained flooding and multiple random walk, no topology reconstruction is required in related simulations. Fig. 4 describes the node degree distribution of the initial simulated P2P network.

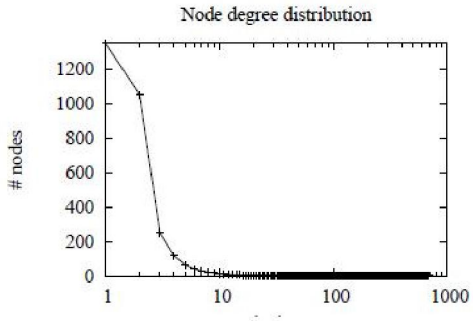

Figure 4 Node degree distribution of the simulated P2P network

We also took into account the impact of the distribution of resource instances, the distribution of queries, and their combinations on the network performance. Hence, we followed the practice in [7] and simulated networks in which the distribution of resource replication and the query distribution may follow one of Gaussian distribution, Zipf distribution, and the discrete uniform distribution, respectively.

Our simulator generated a series of queries at the same rate across all the experiments and randomly selected a peer to be the originator for each query. We set TTL for all queries to 60.

We mainly evaluated the performance of the proposed hybrid search scheme using a number of metrics: the recall, the hops of query results, the number of query messages processed per query, the number of query messages processed per node, the peak number of messages in the message queue amongst all nodes for processing, and the maximum hops of query results. These metrics help capture the fundamental characteristics of the scheme relevant to the comparison with other benchmark methods.

B. Results and Analysis

Table I shows the result of a comparison among the proposed hybrid P2P search, multiple random walk, and

[3] The efficiency of a link was defined as the number of unique queries over the total number of queries received from the link.

Table I A COMPARISON AMONG THE HYBRID P2P SEARCH, MULTIPLE RANDOM WALK (MRW), AND CONSTRAINED FLOODING (C-FLD), NETWORK SIZE = 3037

search method	*1 hop replication*	*result caching*	*# super node*	*# walker*	*recall*	*hops*	*msgs/query*	*msgs/node*	*peak # msgs*	*max hops*
Hybrid Search	û	û	13		0.998	3.601	6520.979	317.782	11008	8
	û	û	102		0.998	3.593	6532.500	318.343	11051	8
	û	û	640		0.998	3.590	6570.121	320.177	11207	8
	û	ü	13		0.998	3.387	6522.331	317.848	10325	8
	ü	ü	13		0.998	2.662	6700.040	326.508	11021	8
Hybrid Search (halve flooding coverage)	ü	ü	13		0.951	2.970	3625.554	176.681	343	11
MRW	û	û		20	0.652	6.051	2389.554	116.448	116	14
	û	û		40	0.757	5.169	2755.074	134.260	792	12
	û	û		80	0.835	4.640	3035.250	147.914	2533	12
	û	û		160	0.897	4.276	3253.783	158.564	6910	11
C-FLD	û	û			1.000	3.591	6666.871	324.891	12295	8

constrained flooding. We observed that one-hop replication, caching history query results, and their combinations help reduce the hops of query result messages (see the 1st, 4th, and 5th rows of results related to the hybrid search) which are the delay of query results perceived by users. Further, by increasing the number of supernodes in the network, the hops of query result messages are also decreased.

However, a large number of messages per query and a large number of messages per node (reflecting part of the system load) are seen with all settings for the hybrid search method. Apart from a much less number of query message hops (2.662 in the hybrid search vs. 3.591 in constrained flooding), the results of all other metrics are comparable to those of constrained flooding. This is mainly due to the extensive number of messages generated during flooding of query messages. We therefore considered decreasing the coverage of flooded messages by only allowing half of the messages to be sent. The experimental result indicates that by doing so the system load can be reduced by around 50% at the cost of a slightly decreased recall level (from 99.8% to 95.1%) and an extended query path (from 2.662 hops to 2.970 hops).

Although multiple random walk possesses good scalability, the major problem with the method is that its search performance cannot be significantly enhanced when the number of walkers reaches a certain threshold. Similar to the findings in [7], we observed that it is very difficult to achieve a recall level close to 100% by using multiple random walk alone.

Recall the rationale of topology reorganization in Sect. III-E that by allowing peer nodes to proactively select more efficient neighbors, clusters that made up of nodes with common service descriptions are formed. Once a node that satisfies an incoming query is discovered, other target nodes can be easily located by allowing message forwarding along efficient links that lead to peers within the same cluster, thus no need for the node to always forward query messages to every neighbor. This was mainly intended to reduce the number of generated messages (per node and per query) resulted from flooding of query messages.

We present the simulation result of performance change introduced by topology reorganization in Table II that corresponds with our anticipation. Reconstructing network topology results in significant reduction of system load. For example, when the resource replication and the query distribution follow the Gaussian and the discrete uniform distributions, respectively, both the *messages per query* and the *messages per node* decrease by 9.7% at the cost of a reduced recall level that drops from 97.6% to 94.3%. However, as can be seen from the result, topology reorganization does not work for all the combinations of the resource replication and the query distribution. In particular, when the resource replication follows the discrete uniform distribution, the metrics including the recall and hops indicate the greatest performance deterioration.

V. CONCLUSIONS AND FUTURE WORK

Drawing on our experience with unstructured P2P, we carried out a preliminary study on Cloud service discovery by adopting an unstructured P2P paradigm. A hybrid search scheme was proposed for service query routing that couples with a number of components including one-hop replication, semantic-aware message routing, topology reorganization, and supernodes for enhanced system performance. We demonstrated, through simulation, that each aforementioned component helps improve the system performance in one way or another whilst a combination of all works the best (still subject to the resource replication distribution and the query distribution as indicated by Table II). We argue that introducing “semantics” to both query message routing and topology reorganization plays an important role in such achievement.

In future work, we will first evaluate the performance of the hybrid search scheme in networks of various

Table II PERFORMANCE CHANGE INTRODUCED BY TOPOLOGY REORGANIZATION

resource replication	*query distribution*	*topology reorganization*	*recall*	*hops*	*msgs/query*	*msgs/node*	*peak # msgs*	*max hops*
Gaussian	uniform	û	0.976	2.929	3583.980	118.010	343	10
		ü	0.943	3.222	3237.360	106.597	302	10
	Zipf	û	0.942	2.831	3524.966	171.779	343	11
		ü	0.892	3.162	3027.310	147.527	285	11
Zipf	uniform	û	0.952	3.080	3598.810	118.498	343	10
		ü	0.929	3.432	3239.210	106.658	304	10
	Zipf	û	0.951	2.970	3625.554	176.681	343	11
		ü	0.924	3.274	3354.486	163.471	320	12
uniform	uniform	û	0.953	3.166	3715.990	122.357	343	11
		ü	0.837	3.478	2817.330	92.766	268	11
	Zipf	û	0.952	2.999	3686.513	179.652	343	11
		ü	0.869	3.415	2941.256	143.334	276	12

topologies and scales. Moreover, we anticipate developing matching and sorting algorithms for semantics-based service matching since it is indispensable for realizing effective and efficient service discovery in the context of Cloud computing.

ACKNOWLEDGMENT

This work is funded by China Postdoctoral Science Foundation (No. 20100470557), the 382 Research Foundation for Talented Scholars (No. G08382320), the Engineering Disciplines Planning Project (No. XNG0921), and the Leading Academic Discipline Program (3rd phase of 211 Project). We also acknowledge the input of the National Natural Science Foundation of China (No.61035003, No. 60933004, No. 61072085, No.60903141, and No.60970088), and National Basic Research Priorities Programme (No. 2007CB311004).

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “Above the clouds: A berkeley view of cloud computing,” Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Tech. Rep. UCB/EECS-2009-28, 2009.

[2] R. Buyya, S. Pandey, and C. Vecchiola, “Cloudbus toolkit for marketoriented cloud computing,” in Proceedings of the 1st International Conference on Cloud Computing, Beijing, China, 2009, pp. 24–44.

[3] R. Ranjan, L. Zhao, X. Wu, and A. Liu, “Peer-to-peer cloud provisioning: Service discovery and load-balancing,” The Computing Research Repository, vol. abs/0912.1905, 2009.

[4] G. S. Manku, M. Naor, and U. Wieder, “Know thy neighbor’s neighbor: the power of lookahead in randomized p2p networks,” in Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, Chicago, USA, 2004, pp. 54–63.

[5] J. Zhou, W. Hall, D. C. D. Roure, and V. K. Dialani, “Supporting ad-hoc resource sharing on the web: A peer-to-peer approach to hypermedia link services,” ACM Transactions on Internet Technology, vol. 7, no. 2, May 2007.

[6] J. Zhou and D. D. Roure, “Floodnet: Coupling adaptive sampling with energy aware routing in a flood warning system,” Journal of Computer Science and Technology, vol. 22, no. 1, pp. 121–130, 2007.

[7] J. Zhou, W. Hall, and D. D. Roure, “Building a distributed infrastructure for scalable triple stores,” Journal of Computer Science and Technology, vol. 24, no. 3, pp. 447–462, 2009.

[8] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energyefficient communication protocol for wireless microsensor networks,” in Proceedings of the 33rd Hawaii International Conference on System Sciences, vol. 8, Jan. 2000, pp. 3005–3014.

[9] P. Ganesan, B. Yang, and H. Garcia-Molina, “One torus to rule them all: multi-dimensional queries in p2p systems,” in Proceedings of the 7th International Workshop on the Web and Databases, Paris, France, 2004, pp. 19–24.

[10] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara, “Owl-s: Semantic markup for web services,” http://www.daml.org/services/owl-s/1.2/, 2010.

[11] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, , and D. Fensel, “Web service modeling ontology,” Applied Ontology, vol. 1, no. 1, pp. 77–106, 2005.

[12] R. Akkiraju, J. Farrell, J. Miller, M. Nagarajan, M.-T. Schmidt, A. Sheth, and K. Verma, “Web service semantics - wsdl-s,” http://www.w3.org/Submission/WSDL-S/, 2005.

[13] A. M. Zaremski and J. M. Wing, “Signature matching: a tool for using software libraries,” ACM Transactions on Software Engineering and Methodology, vol. 4, no. 2, pp. 146–170, 1995.

[14] A. M. Zaremski and J. M. WING, “Specification matching of software components,” ACM Transactions on Software Engineering and Methodology, vol. 6, no. 4, pp. 333–369, 1997.

[15] A. Harth, S. Decker, Y. He, H. Tangmunarunkit, and C. Kesselman, “A semantic matchmaker service on the grid,” in Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, New York, NY, USA, 2004, pp. 326–327.

[16] S. A. Ludwig and S. M. S. Reyhani, “Introduction of semantic matchmaking to grid computing,” Journal of

Parallel and Distributed Computing, vol. 65, no. 12, pp. 1533–1541, 2005.
[17] G. Zhou, J. Yu, R. Chen, and H. Zhang, "Scalable web service discovery on p2p overlay network," in Proceedings of the IEEE International Conference on Services Computing (SCC 2007), Salt Lake City, Utah, USA, 2007, pp. 122–129.
[18] Y. Li, F. Zou, Z. Wu, and F. Ma, "Pwsd: A scalable web service discovery architecture based on peer-to-peer overlay network," in Proceedings of the 6th Asia-Pacific Web Conference, 2004, pp. 291–300.
[19] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara, "Semantic matching of web services capabilities," in Proceedings of the 1st International Semantic Web Conference on The Semantic Web, Sardinia, Italy, 2002, pp. 333–347.
[20] S. B. Mokhtar, D. Preuveneers, N. Georgantas, V. Issarny, and Y. Berbers, "Easy: Efficient semantic service discovery in pervasive computing environments with qos and context support," Journal of Systems and Software, vol. 81, no. 5, pp. 785–808, May 2008.
[21] C. Zeng, X. Guo, W. Ou, and D. Han, "Cloud computing service composition and search based on semantic," in Proceedings of the 1st International Conference on Cloud Computing, 2009, pp. 290–300.
[22] T. J. Lukka and B. Fallenstein, "Freenet-like guids for implementing xanalogical hypertext," in Proceedings of the 13th ACM conference on Hypertext and hypermedia, College Park, Maryland, USA, 2002, pp. 194–195.
[23] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network," in Proceedings of the 2001 ACM SIGCOMM Conference, San Diego, California, USA, 2001, pp. 161–172.
[24] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, H. B. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in Proceedings of the 2001 ACM SIGCOMM Conference, San Diego, California, USA, 2001, pp. 149–160.
[25] A. I. T. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms, Heidelberg, Germany, 2001, pp. 329–350.
[26] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Tech. Rep. UCB/CSD-01-1141, 2001.
[27] H. Samet, The Design and Analysis of Spatial Data Structure. Addison-Wesley Publishing Company, 1990.
[28] M. Cai, M. Frank, J. Chen, and P. Szekely, "Maan: A multi-attribute addressable network for grid information services," Journal of Grid Computing, vol. 2, pp. 3–14, 2004.
[29] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer systems," in Proceedings of the 16th international conference on Supercomputing, New York, New York, USA, 2002, pp. 84–95.
[30] A. Iamnitchi, I. Foster, and D. C. Nurmi, "A peer-to-peer approach to resource location in grid environments," in Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing, 2002, pp. 419.
[31] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutella-like p2p systems scalable," in Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, Karlsruhe, Germany, 2003, pp. 407–418.
[32] C. Gkantsidis, M. Mihail, and A. Saberi, "Hybrid search schemes for unstructured peer-to-peer networks," in Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, Miami, USA, 2005, pp. 1526–1537.
[33] J. Winick and S. Jamin, "Inet-3.0: Internet topology generator," University of Michigan, Technical Report CSE-TR-456-02, 2002.

Jing Zhou received her B.Eng. and M.Eng. degrees in computer science from the School of Informatics at Wuhan Technical University of Surveying and Mapping, P.R. China, in 1997 and 2000, respectively, and obtained a Ph.D. degree in computer science from the University of Southampton, UK, in 2005. She was a postdoctoral research assistant in the School of Electronics and Computer Science at the University of Southampton from 2004 to 2006. She came to Beijing in 2007 to join the School of Computer Science at the Communication University of China and was promoted to Associate Professor in 2008. Dr. Zhou is a member of ACM. Her long-term research interest centers around the development of decentralized algorithms for large scale distributed systems .

Nor Aniza Abdullah received her Bachelor of Computer Science (Hons) degree from the University Malaya, Malaysia in year 1997. In the following year, she obtained her Master of Science degree in the field of interactive multimedia from the University of Westminster, London. In year 2006 she received her Ph.D. degree in Computer Science from the Southampton University, United Kingdom, specializing in adaptive hypermedia research. She started her career as an analyst programmer in an electronic payment system using contactless smartcard in Malaysia. Currently, she is a senior lecturer at the Faculty of Computer Science and Information Technology in University of Malaya, Malaysia. Her research interests revolve around adaptation and personalization of multimedia content, and multimedia content analysis. She has published several numbers of research papers in internationally well known journals such as Journal of Multimedia Tools and Applications, Australasian Journal of Educational Technology and Educational Technology Research and Development.

Zhongzhi Shi, a full professor at the Institute of Computing Technology, Chinese Academy of Sciences, graduated in Computer Science from the Graduate School of University of Science and Technology of China in 1968. He is Director of the Intelligence Science Lab and a PhD supervisor of Ph.D. students at the Institute of Computing Technology, Chinese Academy of Sciences. His research and teaching interests include intelligence science, distributed intelligence, machine learning, neural computing and data mining. He has published 11 monographs, 12 books and more than 400 research papers in journals and conferences. Active in professional activities, Prof. Shi is a senior member of IEEE, ACM and AAAI member, a Chair for the WG 12.2 of IFIP. He serves as a Vice President for Chinese Association of Artificial Intelligence, Executive president of Chinese Neural Network Council.

